memoria

April 4, 2019

Type Package

Title What the Package Does (Title Case)

Version 0.1.0

Author Who wrote it

Maintainer The package maintainer <yourself@somewhere.net>

Description More about what it does (maybe more than one line)
Use four spaces when indenting paragraphs within the Description.

License What license is it under?

Encoding UTF-8

LazyData true

RoxygenNote 6.1.1

R topics documented:

`computeMemory` *Organizes time series data into lags.*

Description

Takes an oputput of `prepareLaggedData` to fit the following model with Random Forest: $p_t\ p_{t-1} + ... + p_{t-n} + d_t + d_{t-1} + ... + d_{t-n}$

where:

- d is a driver (several drivers can be added).

- t is the time of any given value of the response p.
- $t - 1$ is the lag number 1 (in time units).
- $p_{t-1} + ... + p_{t-n}$ represents the endogenous component of ecological memory.
- $d_{t-1} + ... + d_{t-n}$ represents the exogenous component of ecological memory.
- d_t represents the concurrent effect of the driver over the response.

Usage

```
computeMemory = function(lagged.data,
 drivers = NULL,
 response = "Response",
 add.random = TRUE,
 random.mode = "autocorrelated",
 repetitions = 10,
 subset.response = "none"
)
```

Arguments

`lagged.data`	a lagged dataset resulting from `prepareLaggedData`.
`drivers`	a string or vector of strings with variables to be used as predictors in the model (i.e. c("Suitability", "Driver.A"))
`add.random`	if TRUE, adds a random term to the model, useful to assess the significance of the variable importance scores
`drivers`	character vector, names of the numeric columns to be used as predictors in the model.
`random.mode:`	either "white.noise" or "autocorrelated". See details and `addRandomColumn`.
`repetitions:`	integer, number of random forest models to fit
`response:`	character string, name of the response variable (typically, "Response_0")
`subset.response:`	character string with values "up", "down" or "none", triggers the subsetting of the input dataset. "up" only models memory on cases where the response's trend is positive, "down" selectes cases with negative trends, and "none" selects all cases.

Details

The function returns a dataframe. Column names have the lag number as a suffix. The response variable is identified by changing its name to "Response".

Value

A dataframe with columns representing time-delayed values of the drivers and the response.

Author(s)

Blas M. Benito <blasbenito@gmail.com>

See Also

computeMemory

Examples

```
#loading data
data(simulation)

#adding lags
sim.lags <- prepareLaggedData(
  input.data = simulation[[1]],
  response = "Pollen",
  drivers = "Driver",
  time = "Time",
  lags = seq(0, 200, by=20),
  time.zoom=NULL,
  scale=FALSE
  )
```

---

parameters *Parameters of 2 virtual taxa.*

Description

A dataframe with 2 rows and 16 columns with the parameters of two virtual taxa. It was generated by parametersDataframe and fixParametersTypes. It is meant to be used as input for simulatePopulation. It's columns are:

Usage

```
data(drivers)
```

Format

numeric vector of length 10000.

Details

- label: to store names (character string) of the virtual taxa.
- maximum.age: integer, maximum possible age of the individuals in years.
- reproductive.age: integer, age of sexual maturity in years.
- fecundity: integer, number of maximum viable seeds produced by a mature individual under fully suitable conditions.
- growth.rate: numeric, parameter of the logistic growth function.

- pollen.control: numeric in the interval [0, 1]. If 0, pollen productivity depends on environmental suitability only. The larger the number, biomass takes over environmental suitability in determining pollen productivity.
- maximum.biomass: integer, maximum biomass of the individuals.
- carrying.capacity: integer, maximum sum of biomass of the individuals. Very large carrying capacities plus a low maximum.biomass generates too many individuals for the simulation to remain efficient. Try to set carrying.capacity and maximum.biomass to carrying.capacity divided by biomass returns a number lower than 1000 (and even better if it is closer to 100).
- driver.A.weight: numeric in the interval [0, 1], represents the relative influence of the driver on environmental suitability.
- driver.B.weight: numeric in the interval [0, 1], represents the relative influence of the driver on environmental suitability. The sum of weights of drivers A and B should be 1.
- niche.A.mean: numeric, in the same units as driver A. It is the mean of the normal function defining the response of the virtual taxa to driver A.
- niche.A.sd: numeric, in the same units as driver A. It is the standard deviation of the normal function defining the response of the virtual taxa to driver A.
- niche.B.mean: as above, but for driver B.
- niche.B.sd: as above, but for driver B.
- autocorrelation.length.A: numeric, only useful if several drivers generated with different autocorrelation lengths are available (and identified by the column `autocorrelation.length`) in the `drivers` argument provided to the `simulatePopulation` function.
- autocorrelation.length.B: same as above.

Author(s)

Blas M. Benito <blasbenito@gmail.com>

See Also

`parametersCheck`, `parametersDataframe`, `simulatePopulation`

`prepareLaggedData` *Organizes time series data into lags.*

Description

Takes a multivariate time series, where at least one variable is meant to be used as a response in a model while the others are meant to be used as predictors, and organizes it in to quantify ecological memory through models of the form: $p_t \ p_{t-1} + ... + p_{t-n} + d_t + d_{t-1} + ... + d_{t-n}$

where:

- d is a driver (several drivers can be added).
- t is the time of any given value of the response p.

- $t - 1$ is the lag number 1 (in time units).
- $p_{t-1} + ... + p_{t-n}$ represents the endogenous component of ecological memory.
- $d_{t-1} + ... + d_{t-n}$ represents the exogenous component of ecological memory.
- d_t represents the concurrent effect of the driver over the response.

Usage

```
prepareLaggedData(
  input.data = NULL,
  response = NULL,
  drivers = NULL,
  time = NULL,
  lags = seq(0, 200, by=20),
  time.zoom=NULL,
  scale=FALSE
  )
```

Arguments

`input.data`	a dataframe with one time series per column.
`response`	character string, name of the numeric column to be used as response in the model.
`drivers`	character vector, names of the numeric columns to be used as predictors in the model.
`time`	character vector, name of the numeric column with the time/age.
`lags`	numeric vector, lags to be used in the equation. Generally, a regular sequence of numbers. The use `seq` to define it is highly recommended. If 0 is absent from lags, it is added automatically to allow the consideration of a concurrent effect.
`time.zoom`	numeric vector of two numbers of the `time` column used to subset the data if desired.
`scale`	boolean, if TRUE, applies the `scale` function to normalize the data. Required if the lagged data is going to be used to fit linear models.

Details

The function returns a dataframe. Column names have the lag number as a suffix. The response variable is identified by changing its name to "Response".

Value

A dataframe with columns representing time-delayed values of the drivers and the response.

Author(s)

Blas M. Benito <blasbenito@gmail.com>

See Also

computeMemory

Examples

```
#loading data
data(simulation)

#adding lags
sim.lags <- prepareLaggedData(
  input.data = simulation[[1]],
  response = "Pollen",
  drivers = "Driver",
  time = "Time",
  lags = seq(0, 200, by=20),
  time.zoom=NULL,
  scale=FALSE
  )
```

---

simulation *List with two simulation outputs.*

Description

A list of dataframes with two slots, output of simulatePopulation. Each slot corresponds to a virtual taxon from the parameters dataframe. Each dataframe has the following columns:

Usage

```
data(simulation)
```

Format

numeric vector of length 10000.

Details

- *Time* integer, ages in years. Negative ages indicate the burn-in period.
- *Pollen* numeric, pollen counts
- *Population.mature* numeric, number of mature individuals.
- *Population.immatre* numeric, number of immature individuals.
- *Population.viable.seeds* numeric, number of viable seeds generated each year.
- *Suitability* numeric, environmental suitability computed from the driver by the normal function/s defining the taxon niche.
- *Biomass.total* numeric, overall biomass of the population.

- *Biomass.mature* numeric, sum of biomass of mature individuals.
- *Biomass.immature* numeric, sum of biomass of immature individuals.
- *Mortality.mature* numeric, number of mature individuals dead each year.
- *Mortality.immature* numeric, same as above for immature individuals.
- *Driver.A* numeric, values of driver A.
- *Driver.B* numeric, values of driver B, if available, and NA otherwise.
- *Period* qualitative, with value "Burn-in" for burn-in period, and "Simulation" otherwise.

Author(s)

Blas M. Benito <blasbenito@gmail.com>

See Also

simulatePopulation, plotSimulation

`toRegularTime` *Reinterpolates aggregated simulations into regular time.*

Description

Takes the output of aggregateSimulation, and interpolates it into a regular time grid.

Usage

```
toRegularTime(
  x,
  time.column="Time",
  interpolation.interval,
  columns.to.interpolate=c("Suitability", "Driver.A", "Pollen")
  )
```

Arguments

`x`	list of dataframes (generally the output of aggregateSimulation) or single dataframe with irregular time series.
`time.column`	character string, default value is "Time".
`interpolation.interval`	integer, in years, time length encompassed by each sample.
`columns.to.interpolate`	character string or character vector, columns of simulation output to be interpolated. Any subset of: "Pollen", "Population.mature", "Population.immature", "Population.viable.seeds", "Suitability", "Biomass.total", "Biomass.mature", "Biomass.immature", "Mortality.mature", "Mortality.immature", "Driver.A", "Driver.B".

Details

This function fits a loess model of the form y ~ x, where y is any column given by columns.to.interpolate and x is the column given by the time.column argument. The model is used to interpolate column y on a regular time series of intervals equal to interpolation.interval. If x is a matrix-like list returned by aggregateSimulation (on results of simulateAccumulationRate and simulatePopulation), the first column of the matrix will already have a regular time column, and therefore nothing will be done with this column of the list.

Value

If x is a list of dataframes, the function returns a list with the same structure as the input list. If x is a dataframe, the function returns a dataframe. In any case, output dataframes have the columns "Time" (now regular), and any column listed in columns.to.interpolate.

Author(s)

Blas M. Benito <blasbenito@gmail.com>

See Also

simulateAccumulationRate, aggregateSimulation

Examples

```
#loading data
data(simulation)

#generating accumulation rate
acc.rate <- simulateAccumulationRate(
 seed=50,
 time=1:1000,
 output.min=10,
 output.max=40,
 direction=1,
 plot=TRUE
 )

#aggregating simulated data
sim.output.aggregated <- aggregateSimulation(
 simulation.output=sim.output,
 accumulation.rate=acc.rate,
 sampling.intervals=3
 )

#comparing simulations
sim.output.regular <- toRegularTime(
 x=sim.output.aggregated,
 time.column="Time",
 interpolation.interval=20,
 columns.to.interpolate=c("Driver.A", "Pollen")
 )
```

Index